



PATENT ABSTRACTS OF JAPAN

(11) Publication number: **2000057199 A**(43) Date of publication of application: **25 . 02 . 00**

(51) Int. Cl.

G06F 17/50
H01L 21/82
H01L 27/04
H01L 21/822

(21) Application number: **10229840**(71) Applicant: **TOSHIBA CORP**(22) Date of filing: **14 . 08 . 98**(72) Inventor: **IMAI HIROSHI**

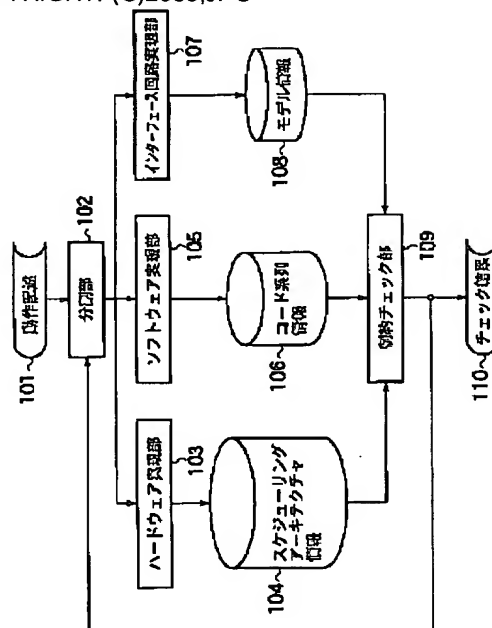
(54) **DEVICE FOR SUPPORTING DESIGN OF SYSTEM
 AND COMPUTER- READABLE RECORDING
 MEDIUM RECORDING DESIGN SUPPORT
 PROGRAM**

COPYRIGHT: (C)2000,JPO

(57) Abstract:

PROBLEM TO BE SOLVED: To make easily and quickly designable more correct system by considering influence of an interface means for transferring data between hardware and software at the time of designing a system by sharing processing functions between the hardware and the software to execute evaluation.

SOLUTION: The system design supporting device shares an operation description 101 describing the processing functions of a system between a hardware realizing part 103 and a software realizing part 105, evaluates the processings shared by the hardware and the software, checks design restriction by a restriction checking part 109 according to the result of the evaluation and designs the system while adjusting the share by a dividing part 102. The evaluation concerning the processing of an interface means for transferring the processing result between the hardware and the software is executed by an interface circuit realizing part 107 at the time of adjustment and the result is reflected.



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-57199

(P2000-57199A)

(43) 公開日 平成12年2月25日 (2000.2.25)

(51) Int.Cl. ⁷	識別記号	F I	テーマコード* (参考)	
G 0 6 F	17/50	G 0 6 F 15/60	6 5 4 R	5 B 0 4 6
H 0 1 L	21/82		6 3 6 G	5 F 0 3 8
	27/04		6 5 4 D	5 F 0 6 4
	21/822	H 0 1 L 21/82	C	
		27/04	U	
審査請求 未請求 請求項の数 2 O L (全 15 頁)				

(21) 出願番号 特願平10-229840

(22) 出願日 平成10年8月14日 (1998.8.14)

(71) 出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72) 発明者 今井 浩史

神奈川県川崎市幸区堀川町580番1号 株式会社東芝半導体システム技術センター内

(74) 代理人 100083806

弁理士 三好 秀和 (外3名)

Fターム(参考) 5B046 AA08 BA02

5F038 CA17 DF14 EZ09 EZ10 EZ20

5F064 AA01 BB01 DD02 DD03 EE02

EE03 EE57 EE58 HH06 HH09

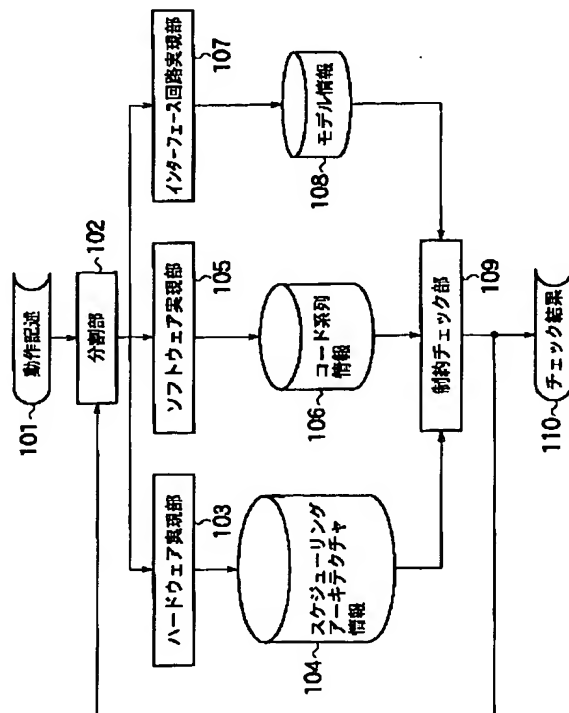
HH11 HH12

(54) 【発明の名称】 システムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体

(57) 【要約】

【課題】 処理機能をハードウェアとソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行う。

【解決手段】 本発明のシステム設計支援装置は、システムの処理機能を記述した動作記述101を、ハードウェア実現部103とソフトウェア実現部105とに分担し、この分担に基づいてハードウェアとソフトウェアによる処理を評価し、この評価の結果によって、制約チェック部109で設計制約をチェックし、これにより分割部102で分担を調整しつつシステムの設計を行うものであって、調整の際に、ハードウェアとソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価をインターフェース回路実現部107で行い、この結果を反映させるものである。



【特許請求の範囲】

【請求項 1】 システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記分担を調整しつつ前記システムの設計を行うシステム設計支援装置であって、前記調整の際に、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価結果を反映させることを特徴とするシステムの設計支援装置。

【請求項 2】 システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記システムの設計を行うシステム設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体であって、システムの処理機能を記述した動作仕様を入力するステップと、前記動作仕様に基づいて前記処理機能をハードウェアとソフトウェアに分担するステップと、前記ハードウェアに分担された動作仕様について評価を行うステップと、前記ソフトウェアに分担された動作仕様について評価を行うステップと、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価を行うステップと、前記ハードウェア、ソフトウェア及びインターフェースについての各評価結果に基づいて、前記システムの設計制約を満たすか否かの判断を行うステップと、前記判断において前記設計制約を満たさない場合には、前記各評価結果に基づいて前記分担を調整するステップとを有することを特徴とするシステム設計支援を記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、コンピュータ等の演算処理システムのハードウェア・ソフトウェア・コードザインに関し、処理動作をハードウェア及びソフトウェアに分割したうえで、これらの処理を評価し、その評価結果が設計制約を満足するかどうかチェックすることによってシステムの設計を支援する装置及びそのプログラムを記録したコンピュータ読み取り可能な記録媒体に関するものである。

【0002】

【従来の技術】近年、ASIC、CPU、メモリ等から構成される複合演算処理システム開発の要求が高まっている。このようなシステムを設計するには、システム全体の処理動作を記述した仕様をASIC等にハードウェア化する部分、CPUで実行されるソフトウェアにする

部分へ分割する必要がある。これをハードウェア・ソフトウェア分割という。

【0003】かかるハードウェア・ソフトウェア分割の際には、ASIC、CPU、メモリ等のシステム全体の面積、性能等のコストを見積もり、設計者が与えた回路規模や性能等の設計制約を満たすように動作仕様をハードウェア・ソフトウェアに分担する必要がある。

【0004】通常、一回の分割処理で制約を満たすハードウェア・ソフトウェア分割が見つかることは希である。

従って、いくつかの分割を試し、設計制約を満たすハードウェア・ソフトウェア分割を見つける。設計制約を満たすハードウェア・ソフトウェア分割を見つけるためには、システム全体の面積、性能等のコストをいかに正確に見積もるかが重要となる。なぜなら、正確に見積もらなかった場合、見積もりの段階では設計制約を満たしていたにもかかわらず、設計段階が進むにつれ実際の値と見積もり値との差が大きくなり、設計制約を満たさないことが分かったときには、分割のやり直しが必要となり、設計期間が長期化する惧れがある。

【0005】さらに、ハードウェア・ソフトウェア分割は複数回行われるので、設計期間を短縮するためには面積、性能等のコストの見積もり時間を短くする必要がある。

【0006】

【発明が解決しようとする課題】しかし、従来、例えばタイミング制約を満足するかどうかチェックする際に、ハードウェアで実現される部分のスケジューリング結果とソフトウェアで実現される部分のソフトウェア規模にのみ基づいてチェックしているため正確なチェックができなかった。従来の見積もり方法については、例えば、R. K. Gupta 著、Co-Synthesis of Hardware and Software for Digital Embedded Systems, Kluwer Academic Publishers, 1995に説明されている。

【0007】また、正確なチェックを行うためには、ASIC等のハードウェアとCPU間のデータの受け渡しに必要なインターフェース回路を設計し、シミュレーションを行う必要があるため、チェックに時間が掛かる。

【0008】一方、設計期間を短縮するため正確さを犠牲にしたチェック方法を採用した場合、設計段階が進み、選択されたハードウェア・ソフトウェア分割に必要なインターフェース回路が設計される段階になると、面積、性能の値と見積もりとの差が大きくなり、意図した回路とは程遠い回路しか設計できないといった問題があった。

【0009】さらには、正確な設計制約を行うため、設計制約チェックにシミュレーションを用いていたのでは、設計制約チェックに時間が掛かり、設計期間が長期化してしまうといった問題があった。

【0010】そこで、本発明は上記事情に鑑みて成されたものであり、その目的は、処理機能をハードウェアと

ソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行うことができるシステムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体を提案することにある。

【0011】

【課題を解決するための手段】上記課題を解決するために請求項1に記載の発明は、システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記分担を調整しつつ前記システムの設計を行うシステム設計支援装置であって、前記調整の際に、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価結果を反映させるものである。

【0012】このような請求項1に記載の発明によれば、ハードウェアとソフトウェアとを分割する際に、これらの間で処理結果の受け渡しを行うインターフェース手段の影響を考慮するため、より正確なシステムの設計を行うことができる。

【0013】また、請求項2に記載の発明は、システムの処理機能をハードウェアとソフトウェアに分担し、この分担に基づいて該ハードウェアと該ソフトウェアによる処理を評価し、この評価の結果によって前記システムの設計を行うシステム設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体であって、システムの処理機能を記述した動作仕様を入力するステップと、前記動作仕様に基づいて前記処理機能をハードウェアとソフトウェアに分担するステップと、前記ハードウェアに分担された動作仕様について評価を行うステップと、前記ソフトウェアに分担された動作仕様について評価を行うステップと、前記ハードウェアと前記ソフトウェアとの間における処理結果の受け渡しを行うインターフェース手段の処理に関する評価を行うステップと、前記ハードウェア、ソフトウェア及びインターフェースについての各評価結果に基づいて、前記システムの設計制約を満たすか否かの判断を行うステップと、前記判断において前記設計制約を満たさない場合には、前記各評価結果に基づいて前記分担を調整するステップとを有するものである。

【0014】このような請求項2に記載の発明によれば、システムの設計を正確にもしくは迅速に行えるという有用な設計支援プログラムの保存、実行、運搬等が容易に行うことができ、より簡便にシステムの設計を行うことができる。

【0015】

【発明の実施の形態】以下、この発明に係るシステムの設計支援装置の実施形態について説明する。

【0016】(1) 実施例1

本発明の第1の実施形態を図面を参照して説明する。図1は本実施形態に係るハードウェア・コデザイン装置の構成図である。同図において、設計対象の動作仕様を記述した動作記述101が、分割部102においてASIC等のハードウェアで実現される部分と、CPU上で実行されるソフトウェアで実現される部分とに分割される。この分割は面積と性能の設計制約を満たす分割が得られるまで様々な分割を試し、繰り返される。

【0017】ハードウェア実現部103では、ハードウェアに分担された動作記述部分のハードウェア実現情報であるスケジューリング・アーキテクチャ情報104が生成される。このスケジューリング情報とは、例えばハードウェアで実現される動作記述部分の実行を完了するまでの時間の情報等であり、また、アーキテクチャ情報とは、ハードウェアの構造、例えば、パイプライン構造等の情報等である。これらの情報からハードウェアへのデータの受け渡しの情報が得られる。

【0018】一方、ソフトウェア実現部105では、CPU上で実行されるソフトウェアとして実現される動作記述部分のコードの規模、実行時間を表すコード系列情報106が生成される。

【0019】他方、インターフェース回路実現部107では、ASIC等で実現されるハードウェアとソフトウェアを実行するCPU間のデータの受け渡しに必要なインターフェース回路を生成する。さらに、このインターフェース回路実現部107は、インターフェース回路の動作の特徴を表すモデル情報108を生成する。

【0020】このモデル情報108は、例えば、ハンドシェイク等のインターフェース・プロトコルのタイプ、データ受け渡しから結果を得るまでの時間、次のデータを受け渡せるまでの待ち時間等である。

【0021】そして、制約チェック部109において、ハードウェアの実現情報であるスケジューリング・アーキテクチャ情報104と、CPU上で実行されるソフトウェアの規模と実行時間の情報であるコード系列情報106と、ハードウェアとCPU間のインターフェース回路の動作の情報であるモデル情報108とに基づいて、ハードウェア・ソフトウェア分割が実行時間等の設計制約を満足するかどうかチェックされ、チェック結果110が出力される。

【0022】このチェック結果110は分割部102へフィードバックされ、ハードウェア・ソフトウェア分割の再試行に利用される。

【0023】このような操作を行うことにより、ハードウェア・ソフトウェア分割の繰り返しの際に、その分割が設計制約を満足するかどうか正確かつ短時間にチェックすることが可能となり、正確な見積もりに基づいたハードウェア・ソフトウェア分割が短時間に得られる。

【0024】(2) 実施形態2

次に本発明の第2の実施形態について説明する。この実施例は第1の実施例で示したハードウェア実現部におけるスケジューリング・アーキテクチャ情報生成を実現するものである。図2にスケジューリング・アーキテクチャ情報生成のフローチャートを示す。

【0025】同図において、ステップ201でASIC等のハードウェアで実現する部分の動作記述を読み込む。次いで、ステップ202でハードウェアのアーキテクチャを選択する。この選択とは、例えば、パイプライン構造を採用する、採用しない等である。

【0026】ステップ203でステップ202で選択されたアーキテクチャに関する情報が生成される。その後、ステップ204で動作記述をハードウェアで実現するために必要な回路部品の種類と個数を決定するアロケーションと呼ばれる処理を実行する。次いで、ステップ205でステップ204で決定した回路部品の種類と個数に基づいて、動作記述をクロック・サイクルごとに実行される動作系列に分割するスケジューリングと呼ばれる処理を実行する。

【0027】ステップ206では、ステップ205で実行されたスケジューリング結果に基づいてスケジューリング情報を生成する。ステップ207でステップ205でスケジューリング処理されたクロック・サイクル毎の各演算を実際の回路部品に割り当てるバインディングと呼ばれる処理を実行する。

【0028】ステップ208でハードウェア記述を生成する。このハードウェア記述に基づいてハードウェアが合成される。動作記述からハードウェア記述を生成する方法は、例えば、D.Gajski他著、High-Level Synthesis: Introduction to Chip and System Design, Kluwer Academic Publishers, 1992が詳しい。

【0029】図3に動作記述の例を示す。また、ハードウェア・ソフトウェア分割の例を図4に示す。この例では、動作記述中の関数 $f(x)$ がハードウェアで実現され、関数 $main()$ がソフトウェアで実現される。また、図3に示した動作記述中の関数 $f(x)$ の動作を表す動作記述部分がステップ201で読み込まれる。

【0030】この動作記述に対してステップ205で実現されるスケジューリング結果の例を図5に示す。この例では、関数 $f(x)$ の実行を完了するのに10クロック・サイクル掛かることが分かる。また、図6に前述したステップ203とステップ206で生成されるスケジューリング・アーキテクチャ情報の例を示す。アーキテクチャ項目は、ハードウェアがパイプライン構造をとるかどうかを表している。

【0031】ここで、パイプライン・ピッチ項目は、各ステージの実行時間 (C) を表す。ステージ数項目は、パイプラインの段数 (N) を表す。レイテンシー項目は、1組のデータが入力されてから結果が得られるまでのクロック・サイクル数 (L) を表す。データ投入サイ

クル項目は、ハードウェアにデータを受け渡せる間隔を表す。この値は、ステージ数とレイテンシーの値から、 L/N で計算できる。

【0032】例えば、図3に示した動作記述に対して、図4に示したハードウェア・ソフトウェア分割が行われた場合で、アーキテクチャとして非パイプラインを選択し、図5に示したスケジューリング結果の場合のスケジューリング・アーキテクチャ情報の例を図7に示す。本実施形態では、アーキテクチャ項目は、非パイプライン構造を選択したので、非パイプラインとなっている。ステージ数項目は、非パイプラインなので1となる。レイテンシー項目は、スケジューリング結果が10クロック・サイクルとなっているので、10となる。

【0033】データ投入サイクル項目は、非パイプラインでハードウェアが構成されるので、1組のデータの処理が終わるまで次のデータをハードウェアに受け渡すことができず、10となる。また、アーキテクチャとして、ステージ数10のパイプラインを選択した場合のスケジューリング・アーキテクチャ情報の例を図8に示す。

【0034】本実施形態では、かかるアーキテクチャ項目は、パイプライン構造を選択したので、パイプラインとなる。また、ステージ数項目は、10ステージのパイプラインを選択したので、10となる。レイテンシー項目は、スケジューリング結果が10クロック・サイクルとなっているので、10となる。データ投入サイクル項目は、10ステージのパイプラインを選択したので、1となる。

【0035】上記のようにスケジューリング・アーキテクチャ情報が作成される。

【0036】(3) 実施形態3

次に本発明の第3の実施形態について説明する。この実施例は第1の実施例で示したソフトウェア実現部におけるコード系列情報生成を実現するものである。図9にコード情報系列情報生成のフローチャートを示す。

【0037】同図において、ステップ901で、ハードウェア・ソフトウェア・コデザインの動作記述中CPU上で実行されるソフトウェアで実現される動作記述部分が読み込まれる。ステップ902で、ステップ901で読み込んだ動作記述を基本命令に分割する。ステップ903で、ステップ902で基本命令に分割された動作記述に基づいてCPU上で実行される基本命令のコード系列を生成する。ステップ904で、ステップ903で生成されたコード系列に基づいてコード系列情報が生成される。ステップ905で、CPU上で実行されるソフトウェアが生成される。動作記述からソフトウェアを生成する方法は、R.Gupta著、Co-Synthesis of Hardware and Software for Digital Embedded Systems, Kluwer Academic Publishers, 1995が詳しい。

【0038】前述した図3に示した動作記述の例に対し

て図4に示したハードウェア・ソフトウェア分割が行われた場合のソフトウェアで実現される関数main()のコード系列の例を図10に示す。図11にステップ904で生成されるコード系列情報の例を示す。コード系列情報として、CPU上でこのソフトウェアを実行するために必要な時間を表すステップ数とハードウェアで実現された関数の呼び出し回数が生成される。

【0039】また、図10に示したコード系列の例に対するコード系列情報の例を図12に示す。図10の例では、ハードウェアで実現される関数f(x)の呼び出しが2回あるので、関数呼び出し回数欄が2となっている。

【0040】(4) 実施形態4

次に本発明の第4の実施形態について説明する。この実施形態では第1の実施例で示したインターフェース回路実現部におけるモデル情報生成を実現するものである。図13にASIC等のハードウェアとソフトウェアを実行するCPU間に必要なインターフェース回路のモデル情報生成のフローチャートを示す。

【0041】同図において、ステップ1301でインターフェース回路情報を読み込む。ステップ1302でステップ1301で読み込んだインターフェース回路情報からインターフェース回路のモデル情報を生成する。ステップ1303でインターフェース回路の動作を表す記述が生成される。この記述が合成され、インターフェース回路が実現される。

【0042】図14にモデル情報の例を示す。この情報は、インターフェース回路のプロトコル・タイプ、データを渡してから結果が得られるまでの時間、データ投入間隔を表すデータ投入サイクルからなる。プロトコル・タイプとしては、例えば、ハンドシェイク、キュー等がある。データを渡してから結果が得られるまでの時間は、ハードウェアの実行時間であり、アーキテクチャ・スケジューリング情報のレイテンシー項目のLとなる。データ投入サイクルは、ハンドシェイク・タイプのインターフェースの場合、1度結果が得られるまで次のデータを渡せないで、Lとなる。

【0043】キュー・タイプのインターフェースの場合、データをインターフェース回路で保持しておくことができるため、結果が得られるまで待つ必要はなく、1となる。図3に示した動作記述に対して図4に示したハードウェア分割が行われた場合のモデル情報の例を図15に示す。ハードウェアがパイプラインで実現されているかいないかに関わらず、結果が得られるまでの時間は10、データ投入サイクルは、ハンドシェイク・タイプの場合は10、キュー・タイプの場合は1となる。

【0044】上記のようにインターフェースのモデル情報が生成される。

【0045】(5) 実施形態5

次に本発明の実施形態について説明する。この実施形態

では第1の実施例で示した制約チェック部におけるハードウェア・ソフトウェア分割が設計制約を満たすかどうかの判断を実現するものである。図16に制約チェックのフローチャートを示す。

【0046】同図において、ステップ1601でハードウェア・ソフトウェア・コデザイン的设计制約を読み込む。ステップ1602でハードウェア実現部102が生成したスケジューリング・アーキテクチャ情報103を読み込む。ステップ1603でソフトウェア実現部104が生成したコード尾系列情報105を読み込む。ステップ1604でインターフェース回路実現部106が生成したモデル情報107を読み込む。ステップ1605でステップ1602から1604で読み込んだ情報に基づいて制約の評価値を計算する。

【0047】ステップ1606でステップ1605で計算した評価値とステップ1601で読み込んだ設計制約を比較する。評価値が設計制約を満たす場合、ステップ1607で設計制約を満たしたことを表示する。評価値が設計制約を満たさない場合、ステップ1608で設計制約を満たさないことを表示する。

【0048】前述した図3に示した動作記述に対して図4に示したハードウェア・ソフトウェア分割が行われ、インターフェース回路としてキュー・タイプのインターフェース回路を仮定した場合のタイミング制約評価値の計算について説明する。

【0049】ここで、ソフトウェアを実行するCPUのクロック・サイクルをC₁、関数f(x)を実現するハードウェアのクロック・サイクルをC₂とする。また、C₁<C₂と仮定する。関数main()中で、xとyの値はハードウェアが計算結果を返すのに必要な時間、すなわち、C₂×Lより十分後に参照されると仮定する。

【0050】まず、ソフトウェアの実行にかかる時間は、図12に示したコード系列情報から選れるステップ数nとCPUのクロック・サイクルC₁との積C₁×nで与えられる。次に、関数f(x)の呼び出しに必要な時間を計算する。図12に示したコード系列情報から、関数f(x)の呼び出しは2回であることが分かる。さらに、2回のf(x)の呼び出しは連続しており、その呼び出し間に待ち時間が必要である。

【0051】この時間を図15に示したインターフェース回路のモデル情報から計算する。インターフェースのプロトコルがハンドシェイク・タイプの場合、モデル情報のデータ投入サイクル項目の値10とハードウェアのクロック・サイクルC₂とから、この時間はC₂×10と計算される。インターフェースのプロトコルがキュー・タイプの場合、モデル情報のデータ投入サイクル項目の値1とハードウェアのクロック・サイクルC₂とから、この時間はC₂×1と計算される。

【0052】本実施形態では、インターフェースのプロトコルとしてキュー・タイプを仮定しており、また、C

C_2 なので、2回の関数 $f(x)$ の呼び出し間の時間を考慮する必要はない。したがって、タイミング制約の評価値は図 17 に示した値となる。なお、従来のように、インターフェース回路のモデルを考慮しない場合、2回の関数 $f(x)$ の呼び出し間の時間を $C_2 \times L$ と評価してしまい、タイミング制約の評価値は図 18 に示すような結果となってしまい、正確なチェックができない。上記のようにハードウェア・ソフトウェア分割が設計制約を満たすかどうかチェックされる。

【0053】さらに、本発明に係るシステムの設計支援装置の他の実施形態について説明する。

【0054】(6) 実施形態 6

本発明の第 6 の実施形態を図面を参照して説明する。図 19 に本実施形態に係るハードウェア／ソフトウェア・コデザイン装置の構成図を示す。

【0055】同図において、ハードウェア／ソフトウェア分割部 1901 で、入力された動作仕様が ASIC 等のハードウェアで実現される部分と、CPU で実行されるソフトウェアで実現される部分とに分割される。この分割は面積と性能の制約を満たす分割が得られるまで繰

り返される。

【0056】また、インターフェース回路詳細化部 1902 では、ハードウェアとソフトウェアを実行する CPU 間等のデータの受渡しのために必要なインターフェース回路を詳細化する。この詳細化の処理は分割探索の期間でただ一度だけ実行される。

【0057】差分情報生成部 1903 では、分割探索期間を通じて、インターフェース回路を詳細化した分割と現分割との差分情報を生成する。この差分情報は現分割に必要となるインターフェース回路の面積と性能を見積

もる際に利用される。

【0058】評価部 1904 では、前記インターフェース回路詳細化部 1902 で詳細化されたインターフェース回路と、前記差分情報生成部 1903 で生成された差分情報に基づいて、現分割の面積と性能の見積もりを実行する。この見積もりには、現分割に必要なインターフェース回路の面積と性能の見積もりをも含む。

【0059】前記ハードウェア／ソフトウェア分割部 1901 は、前記評価部 1904 が生成した見積もりに従って、最終的に最適な分割を選択し、ハードウェア仕様、ソフトウェア仕様、インターフェース仕様を出力する。

【0060】このような操作を行うことにより、最適な分割の探索期間の各分割に対して面積と性能を見積もる際に、正確な見積もりが短期間に可能となり、正確な見積もりに基づいたハードウェア／ソフトウェア分割が得られる。

【0061】(7) 実施形態 7

次に本発明の第 7 の実施形態例について説明する。この実施形態では第 6 の実施形態で示したインターフェース

回路詳細化部におけるインターフェース回路の詳細化を実現するものである。

【0062】動作仕様の例を図 20 に示す。この動作仕様に対しハードウェア／ソフトウェア分割部 1901 に於いて、1つのハードウェア／ソフトウェア分割が生成される。そのアルゴリズムは、例えば、R. K. Gupta, "Co-Synthesis of Hardware and Software for Digital Embedded Systems," Kluwer Academic Publishers, 1995. に見つけられる。

【0063】ハードウェア／ソフトウェア分割の例を図 21 に示す。この例では、図 20 に示した動作仕様中、関数 $f(a, b)$ で表された部分がハードウェアで実現され、動作仕様の残りの部分が CPU 上で実行されるソフトウェアで実現される。

【0064】ハードウェアで実現される動作仕様部分は、例えば、高位合成アルゴリズムによって実現される。高位合成アルゴリズムについては、D. D. Gajski, et al, "High-Level Synthesis: Introduction to Chip and System Design," Kluwer Academic Publishers, 1992. に見つけられる。

【0065】図 21 に示したハードウェア／ソフトウェア分割候補の場合、ハードウェアで実現される $f(a, b)$ のハードウェアの動作を図 22 に示す。図 22 では、ハードウェアの動作をデータ・フロー・グラフ (DFG) で表しており、横線で区切られ、番号付けられた各ステップはステートと呼ばれ、横線はクロック・サイクルの境界を表す。

【0066】同図において、各ステートはクロックに従って順に実行される。DFG 中マルは演算を表し、黒く塗りつぶされた矩形はレジスタを表す。 $f(a, b)$ を実現したハードウェアは、3クロック・サイクル掛かって動作を完了する。さらに、CPU がハードウェアへデータを受け渡すタイミング、CPU がハードウェアからデータを受け取るタイミングを示した。

【0067】図 22 に示した例では、変数 a に対応したデータが CPU からステート 1 に受け渡され、ハードウェアが変数 a にステート 1 でデータを読み込んでいる。なお、変数 b についても同様である。変数 out の値がハードウェアからステート 3 で出力され、変数 out に対応したデータを CPU はステート 3 で読み込んでいる。

【0068】また、インターフェース回路詳細化部 1902 では、インターフェース回路詳細化が実行され、インターフェース回路詳細化情報テーブルが生成される。このインターフェース回路詳細化情報テーブル作成のフローチャートを図 23 に示す。

【0069】初めに、ステップ 2301 と 2302 で、ハードウェアで実現される動作仕様の入力変数と出力変数のリストを作成する。次に、ステップ 2303 で入力変数リスト L_I から変数を 1 つ取り出す。その変数を v とする。ステップ 2304 でその変数 v を取り付けるポ

ート番号を選択する。

【0070】この選択は、自動的に行っても、設計者が自ら手動で行ってもよい。選択されたポート番号がインターフェース回路詳細化情報テーブルに登録されていない場合、まず、ステップ2305でそのポート番号をテーブルに登録する。さらに、ステップ2306から2313で、種類欄、ビット幅欄、ステート欄、バッファ欄、変数欄に、それぞれ、“入力”、変数 v のビット幅、変数 v が読み込まれるステート番号、バッファの有無、変数名 v を登録する。

【0071】選択されたポート番号が既にテーブルに登録されている場合、変数 v のビット幅とテーブルのビット幅欄の値を比較して、変数 v のビット幅の方が大きい場合、ステップ2315で変数 v のビット幅でビット幅欄を更新する。そして、変数 v が読み込まれるステートと変数 v を、それぞれ、ステップ2314と2316でテーブルのステート欄と変数欄に追加する。なお、出力変数リストについても同様な処理を行う。図22に示した例に対するインターフェース回路詳細化情報テーブルを図25に示す。

【0072】次に、前述したステップ2310で実行されるバッファの有無の判断の処理について説明する。この判断のフローチャートを図24に示す。まず、ステップ2401で対象となっている変数が入力変数か出力変数か調べる。対象となる変数が入力変数であった場合、ステップ2402でCPUから変数に対応したデータが与えられるステート(ST i)を調べる。次に、ステップ2403でハードウェア内で変数にデータが読み込まれるステート(ST j)を調べる。

【0073】このとき、対象となる変数が出力変数であった場合、ステップ2404でCPUが変数に対応したデータを読み込むステート(ST i)を調べる。次に、ステップ2405でハードウェア内で変数にデータが書き込まれるステート(ST j)を調べる。ステップ2406でST j が等しいか等しくないかチェックする。ST i とST j が等しい場合、バッファは必要無いので、ステップ2407で、インターフェース回路詳細化情報テーブルのバッファ欄に“無”を登録する。

【0074】ST i とST j が等しくない場合、データを保持しておく必要があるため、ステップ2408で、テーブルのバッファ欄に“有”を登録する。

【0075】上記のような処理により、インターフェース回路詳細化情報テーブルが得られる。このテーブルよりインターフェース回路の詳細な情報が得られる。図25に示した例では、インターフェース回路が2個の入力ポートと1個の出力ポートを持ち、それぞれのポートのビット幅は16で、バッファは必要ないことがわかる。

【0076】上記のようにして、1つのハードウェア／ソフトウェア分割候補のインターフェース回路の詳細化が得られる。

【0077】(8)実施形態8

次に、本発明の第8の実施形態について説明する。この実施形態では第6の実施例で示した差分情報生成部1903で実行される差分情報生成を実現するものである。

【0078】図20に示した動作仕様の図21に示したハードウェア／ソフトウェア分割と異なるハードウェア／ソフトウェア分割を図26に示す。この分割では、動作仕様中 $f(a, b)$ と $g(c, d, e)$ がハードウェアで実現され、動作仕様の残りの部分はCPU上で実行されるソフトウェアで実現される。

【0079】ハードウェアで実現される $f(a, b)$ と $g(c, d, e)$ の動作を図27に示す。 $f(a, b)$ の動作は第7の実施形態での動作と同じである。 g

(c, d, e)に対しては、これを実現するハードウェアは5クロック・サイクル掛かって動作を完了する。データの受渡しのタイミングについては、変数 c に対応したデータがCPUからステート1に受け渡され、ハードウェアが変数 c にステート1でデータを読み込んでい

る。変数 d についても同様である。

【0080】一方、変数 e に関しては、変数 e に対応したデータがCPUからステート1で受け渡されるが、ハードウェアは変数 e にステート2でデータを読み込んでい

る。変数 out の値がハードウェアからステート5に出力され、変数 out に対応したデータをCPUはステート5で読み込んでい

る。

【0081】差分情報生成部1903では、インターフェース回路詳細化部1902が生成したインターフェース回路詳細化情報に基づいて、ハードウェア／ソフトウェア分割に必要なインターフェース回路の差分情報を生成する。

【0082】この差分情報生成のフローチャートを図28に示す。初めに、ステップ2801で入力変数のリストLIを作成し、ステップ2802で入力ポートのリストPIを作成する。ステップ2803で変数を1つリストLIから取り出す。ステップ2804でポートを1つリストPIから取り出す。ステップ2805で変数にバッファが必要かどうか調べる。

【0083】このとき必要な場合、ステップ2804で取り出したポートがバッファを持つかどうか調べる。持っていない場合、新たなポートをリストPIから取り出す。ポートがバッファを持っている場合、ステップ2807で対象となっている変数に対応したデータがCPUから受け渡されるステートST1を調べる。

【0084】ステップ2808でポートをステートST i で使用可能かどうか調べる。使用可能な場合、ステップ2809で使用状況テーブルのステートST i 欄に変数を登録する。

【0085】使用状況テーブルの例を図29に示す。使用不可能な場合、新たなポートをリストPIから取り出し、ステップ2804から新たに処理を始める。リスト

PI から新たなポートが得られなかった場合、ステップ 2810 で差分情報テーブルの入力ポート行の個数欄の値を 1 だけインクリメントする。

【0086】差分情報テーブルの例を図 30 に示す。ステップ 2811 で変数にバッファが必要か調べ、バッファが必要な場合、ステップ 2812 で差分情報テーブルの入力ポート行のバッファ欄の値を 1 だけインクリメントする。ステップ 2813 で使用状況テーブルにポート行を 1 つ追加し、ステップ 2814 でその行の種類欄にを登録し、ステップ 2815 でステート STi 欄に

変数を登録する。上記と同様な処理を出力変数に対しても行う。

【0087】上記のようにハードウェア/ソフトウェア

分割の差分情報が生成される。

【0088】(9) 実施形態 9
次に、本発明の第 9 の実施形態について説明する。この実施形態では第 6 の実施例で示した評価部 1904 で実行される差分情報に基づいたハードウェア/ソフトウェア分割の評価を実現するものである。

【0089】図 21 に示したハードウェア/ソフトウェア分割で必要とされるインターフェース回路と、図 26 に示したハードウェア/ソフトウェア分割で必要とされるインターフェース回路の差分情報を図 30 に示す。この例では、図 26 に示したハードウェア/ソフトウェア分割に必要なインターフェース回路が、入力ポートを 1 個、バッファを 1 個余分に必要であることを示している。

【0090】図 30 に示した差分情報からインターフェース回路の面積評価のフローチャートを図 31 に示す。初めにステップ 3101 でインターフェース回路詳細化部 1902 が生成したインターフェース回路詳細化情報を読み込む。ステップ 3102 でこの情報に基づいて詳細化したインターフェース回路の面積 A をインターフェース回路詳細化情報テーブルから得られる入力ポート数、出力ポート数、バッファ数とあらかじめ与えられているポート、バッファに面積に基づいて計算する。

【0091】次に、ステップ 3103 で差分情報生成部 1903 が生成した差分情報を読み込む。ステップ 3104 で差分情報テーブルの入力ポート行の個数欄の値 (NI) を読み込む。ステップ 3105 で差分情報テーブルの入力ポート行のバッファ欄の値 (BI) を読み込む。ステップ 3106 で差分情報テーブルの出力ポート行の個数欄の値 (NO) を読み込む。ステップ 3107 で差分情報テーブルの出力ポート行のバッファ欄の値 (BO) を読み込む。

【0092】これらの値は詳細化したインターフェース回路と現在評価対象となっているハードウェア/ソフトウェア分割に必要なインターフェース回路の入力ポート数、入力ポートのバッファ数、出力ポート数、出力ポートのバッファ数との差を表している。

【0093】ステップ 3108 で上記 4 ステップで得られた入力ポート数の変化 NI、出力ポート数の変化 NO、バッファ数の変化 BI + BO とあらかじめ与えられているポートとバッファの面積に基づいて、詳細化されたインターフェース回路と現在評価対象となっているハードウェア/ソフトウェア分割に必要なインターフェース回路との面積の差 dA を計算する。ステップ 3109 でステップ 3102 で計算した面積 A と、ステップ 3108 で計算した面積の差 dA とから現在評価対象となっているハードウェア/ソフトウェア分割に必要なインターフェース回路の面積 A + dA を計算する。

【0094】上記のようにハードウェア/ソフトウェア分割に必要なインターフェース回路の面積を、差分情報に基づいて見積もることにより、より正確な評価が短時間に可能となる。

【0095】なお、上述した実施形態で述べた処理手順は、プログラムとして表現することができ、これをコンピュータ読み取り可能な記録媒体に記録することができる。この記録媒体とは、メモリ装置、磁気ディスク装置、光ディスク装置等、プログラムを記録することができるような装置が含まれる。

【0096】

【発明の効果】本発明に係るシステムの設計支援装置及び設計支援プログラムを記録したコンピュータ読み取り可能な記録媒体によれば、処理機能をハードウェアとソフトウェアに分担して評価を行うことによってシステムの設計を行う際に、ハードウェアとソフトウェア間でデータの受け渡しを行うインターフェース手段による影響を考慮することによって、より正確なシステムの設計を容易且つ迅速に行うことができる。

【図面の簡単な説明】

【図 1】本発明の第 1 の実施形態に係るハードウェア・コデザイン装置の構成図である。

【図 2】第 2 の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報生成のフローチャートを示すものである。

【図 3】第 2 の実施形態に係るハードウェア・コデザイン装置における動作記述の例を示すものである。

【図 4】第 2 の実施形態に係るハードウェア・コデザイン装置におけるハードウェア・ソフトウェア分割の例を示すものである。

【図 5】第 2 の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング結果の例を示すものである。

【図 6】第 2 の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例 1 を示すものである。

【図 7】第 2 の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例 2 を示すものである。

【図 8】第 2 の実施形態に係るハードウェア・コデザイン装置におけるスケジューリング・アーキテクチャ情報の例 3 を示すものである。

【図 9】第 3 の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報生成のフローチャートを示すものである。

【図 10】第 3 の実施形態に係るハードウェア・コデザイン装置におけるコード系列の例を示すものである。

【図 11】第 3 の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報の例 1 を示すものである。

【図 12】第 3 の実施形態に係るハードウェア・コデザイン装置におけるコード系列情報の例 2 を示すものである。

【図 13】第 4 の実施形態に係るハードウェア・コデザイン装置におけるモデル情報生成のフローチャートを示すものである。

【図 14】第 4 の実施形態に係るハードウェア・コデザイン装置におけるモデル情報の例 1 を示すものである。

【図 15】第 4 の実施形態に係るハードウェア・コデザイン装置におけるモデル情報の例 2 を示すものである。

【図 16】第 5 の実施形態に係るハードウェア・コデザイン装置における制約チェックのフローチャートを示すものである。

【図 17】第 5 の実施形態に係るハードウェア・コデザイン装置における制約評価値の計算例を示すものである。

【図 18】第 5 の実施形態に係るハードウェア・コデザイン装置における従来の制約評価値の計算例を示すものである。

【図 19】第 6 の実施形態に係るハードウェア・コデザイン装置におけるの構成図である。

【図 20】第 7 の実施形態に係るハードウェア・コデザイン装置における動作仕様の例を示すものである。

* 【図 21】第 7 の実施形態に係るハードウェア・コデザイン装置におけるハードウェア／ソフトウェア分割の例 1 を示すものである。

【図 22】第 7 の実施形態に係るハードウェア・コデザイン装置における図 21 中のハードウェアの動作を示すものである。

【図 23】第 7 の実施形態に係るハードウェア・コデザイン装置におけるインターフェース回路詳細化フローチャートを示すものである。

【図 24】第 7 の実施形態に係るハードウェア・コデザイン装置におけるバッファの有無を判断するフローチャートを示すものである。

【図 25】第 7 の実施形態に係るハードウェア・コデザイン装置におけるインターフェース回路詳細化情報テーブルの例を示すものである。

【図 26】第 8 の実施形態に係るハードウェア・コデザイン装置におけるハードウェア／ソフトウェア分割の例 2 を示すものである。

【図 27】第 8 の実施形態に係るハードウェア・コデザイン装置における図 26 中のハードウェアの動作を示すものである。

【図 28】第 8 の実施形態に係るハードウェア・コデザイン装置における差分情報生成のフローチャートを示すものである。

【図 29】第 8 の実施形態に係るハードウェア・コデザイン装置における使用状況テーブルの例を示すものである。

【図 30】第 9 の実施形態に係るハードウェア・コデザイン装置における差分情報テーブルの例を示すものである。

【図 31】第 9 の実施形態に係るハードウェア・コデザイン装置における評価のフローチャートを示すものである。

* 【符号の説明】

【図 3】

```
int f (int x)
{
  int result=0;
  for (int i=1; i<=10; i++)
    result=result+i*x;
  return result;
}
```

ハードウェア	f (x)
ソフトウェア	main()

【図 11】

```
main ()
{
  int a,b;
  int x,y,z;
```

ステップ数	n
関数呼び出し数	n

```
a=...
b=...
x=f(a);
y=f(b);
...
z=x+y;
}
```

【図 6】

アーキテクチャ	パイプライン
パイプライン・ピッチ	C
ステージ数	N
レイテンシー	L
データ投入サイクル	L/N

【図 12】

ステップ数	n
関数呼び出し数	2

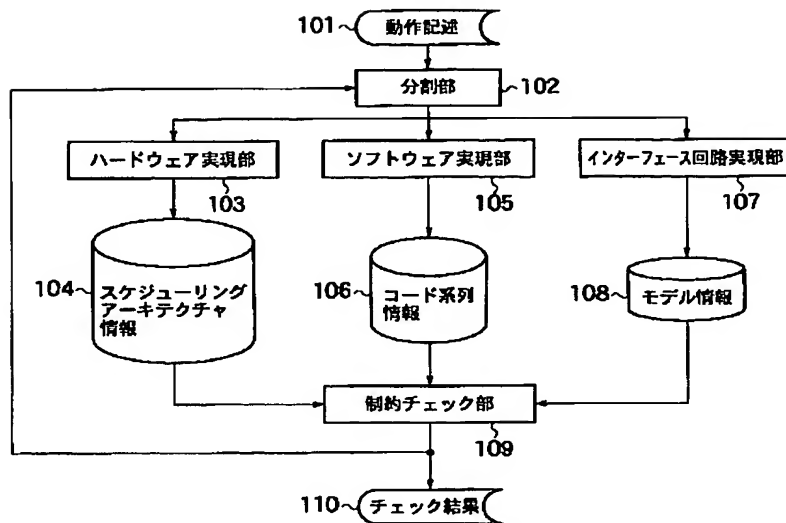
【図 10】 【図 17】

```
... 実行時間=C1×n
add a,...
add b,...
call f(a)
call f(b)
...
ld x
ld y
...
```

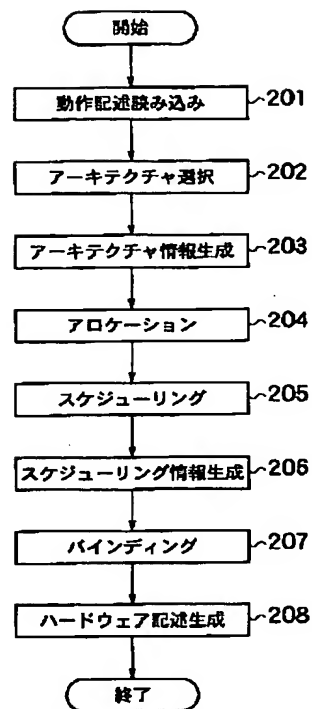
【図 18】

実行時間=C₁×n+C₂×L
=C₁×n+C₂×10

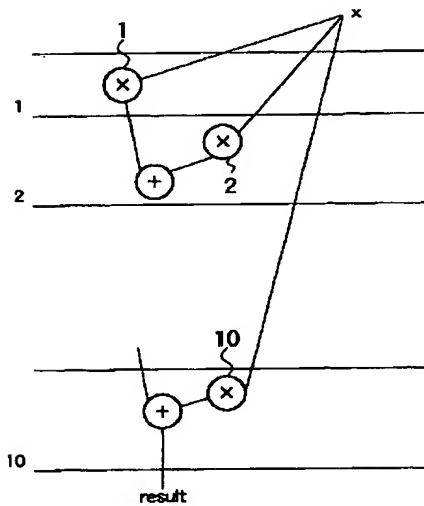
【図 1】



【図 2】



【図 5】



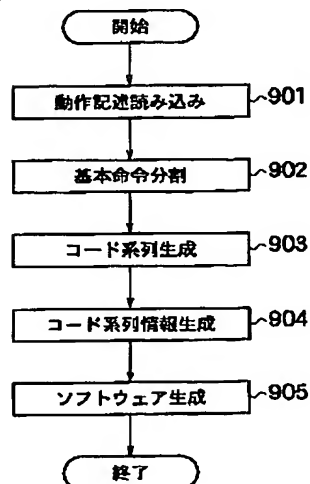
【図 7】

アーキテクチャ	非パイプライン
パイプライン・ピッチ	C
ステージ数	1
レイテンシー	10
データ投入サイクル	10

【図 8】

アーキテクチャ	パイプライン
パイプライン・ピッチ	C
ステージ数	10
レイテンシー	10
データ投入サイクル	1

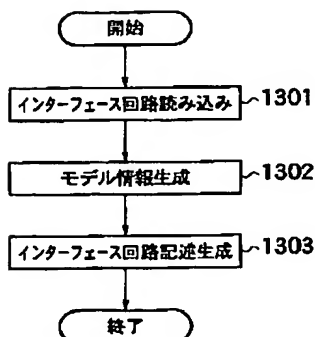
【図 9】



【図 14】

プロトコル・タイプ	ハンドシェイク	キュー
結果が得られるまでの時間	L	L
データ投入サイクル	L	1

【図 13】



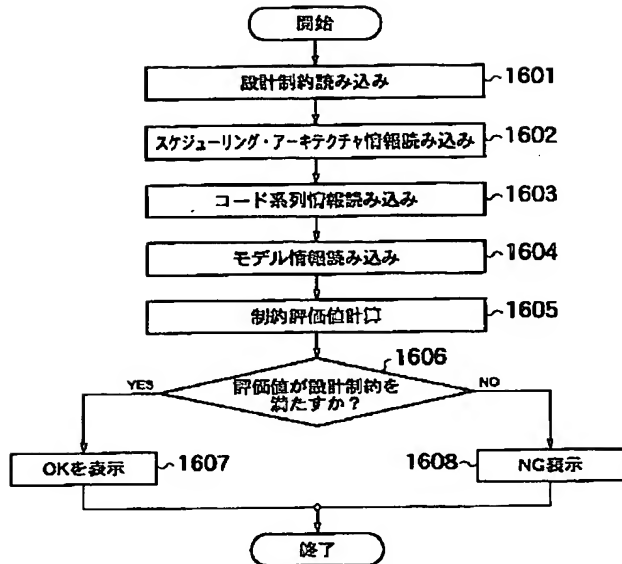
【図 15】

プロトコル・タイプ	ハンドシェイク	キュー
結果が得られるまでの時間	10	10
データ投入サイクル	10	1

【図 21】

ハードウェア	$f(a, b)$
ソフトウェア	動作仕様の残りの部分

【図 16】



【図 20】

```

main()
{
  int a,b,c,d,e;
  int x,y,z;
  :
  x=f(a,b);
  :
  y=g(c,d,e);
  :
  z=x+y;
  :
}

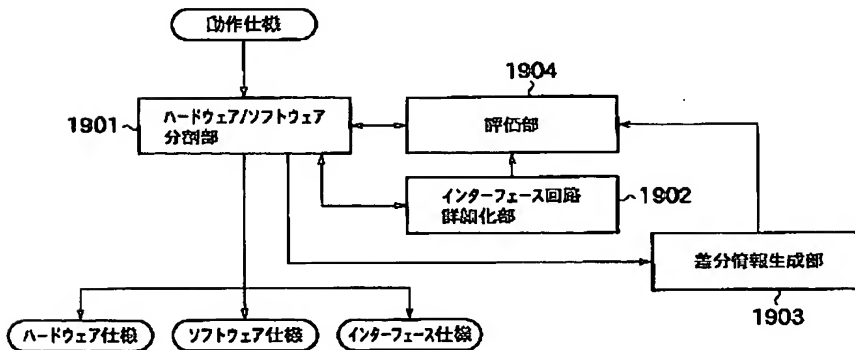
f(a,b)
{
  int out;
  out=a+b;
  return out;
}

g(c,d,e)
{
  int out;
  out=(c*d)-(d*e);
  return out;
}
  
```

【図 25】

ポート番号	種類	ビット幅	タイミング	バッファ	変数
1	入力	16	ステート1	無	a
2	入力	16	ステート1	無	b
3	出力	16	ステート3	無	out

【図 19】



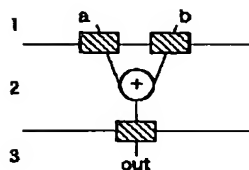
【図 26】

ハードウェア	f(a,b) g(c,d,e)
ソフトウェア	動作仕様の残りの部分

【図 30】

	個数	バッファ
入力ポート	+1	+1
出力ポート	±0	±0

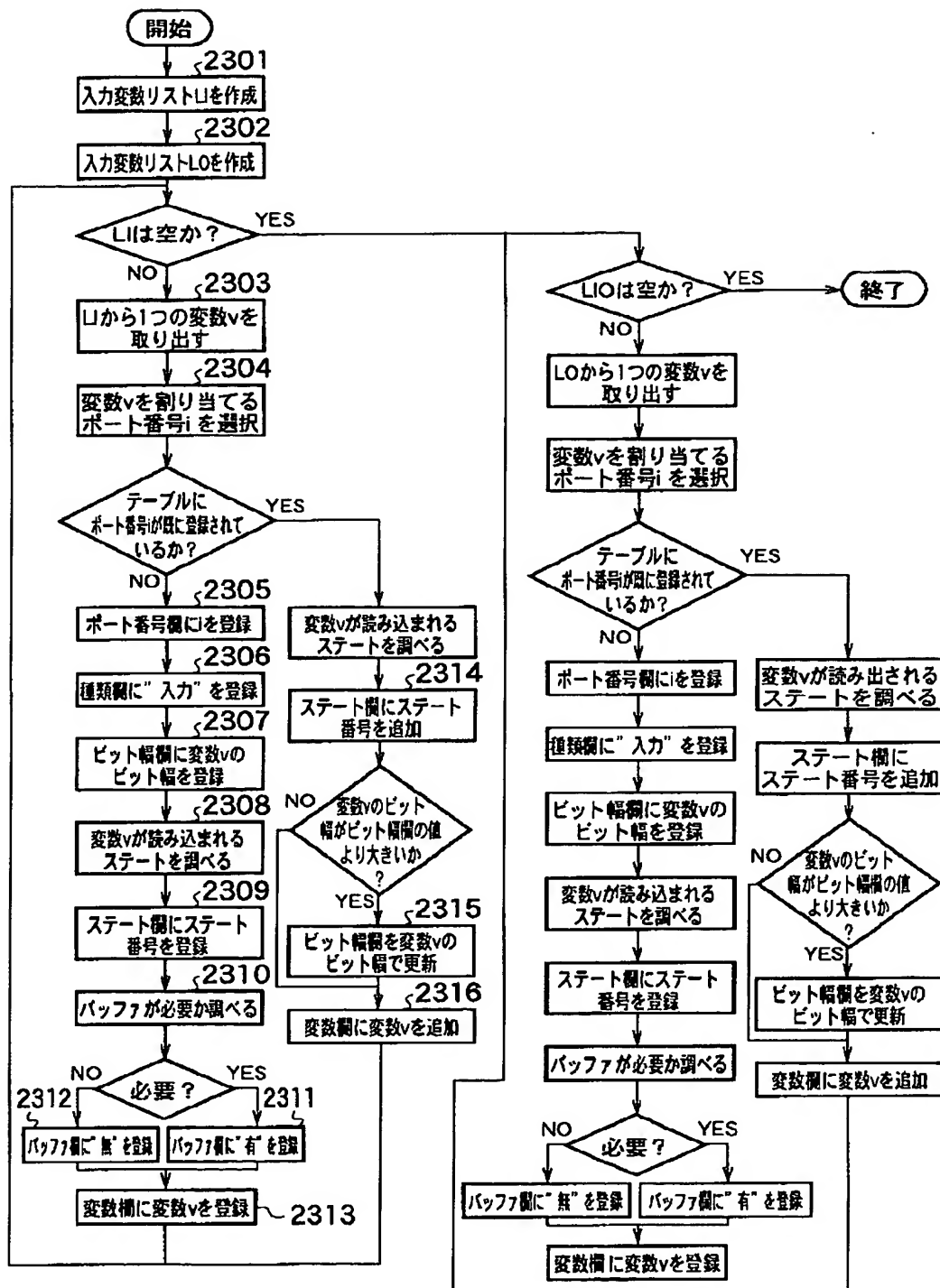
【図 22】



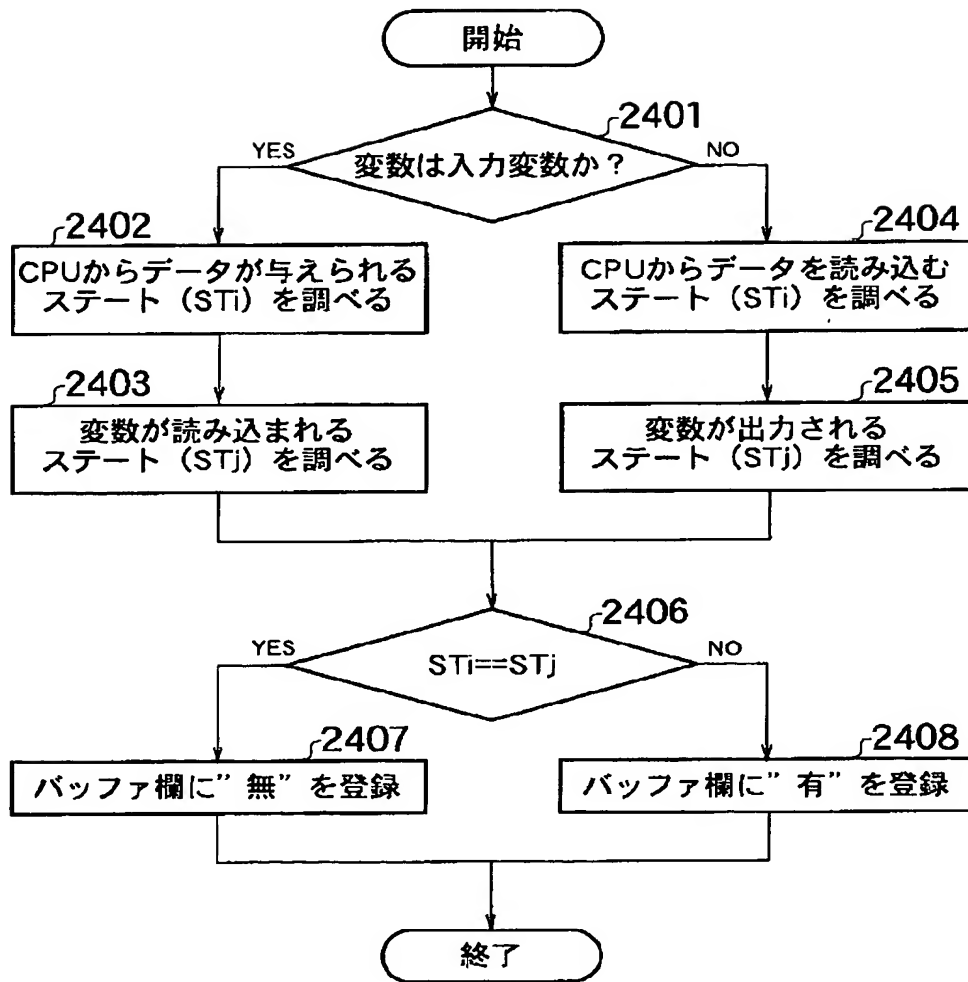
○ 演算
 ▨ レジスタ

変数	CPUから (へ) データが出力 (入力) されるタイミング
a	ステート1
b	ステート1
out	ステート3

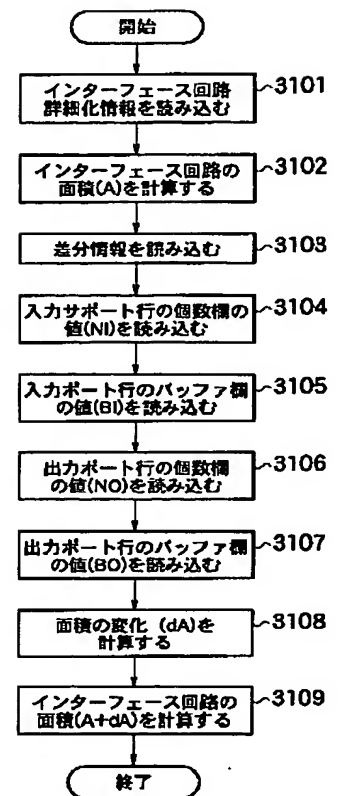
【図 23】



【図 24】



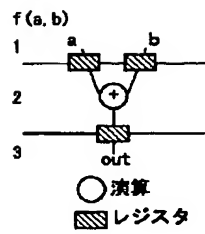
【図 31】



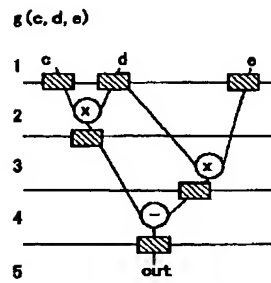
【図 29】

ポート番号	種類	ステート1	ステート2	ステート3	ステート4	ステート5
1	入力	c				
2	入力	d				
3	出力					out
4	入力	e				

【図 27】



変数	CPUから（へ）データが出力（入力）されるタイミング
a	ステート1
b	ステート1
out	ステート3



変数	CPUから（へ）データが出力（入力）されるタイミング
c	ステート1
d	ステート1
e	ステート1
out	ステート4

【図 28】

